EE318062105US

AN INTERACTIVE DATA PROCESSOR CONTROLLED DISPLAY INTERFACE FOR TRACKING OF ALLOCATED MESSAGES IN A DYNAMIC WORKLOAD BALANCING COMMUNICATION SYSTEM

Technical Field

The present invention relates to dynamic workload balancing and distribution in message driven transaction environments, and particularly to the tracking of dynamically allocated messages through a display interface.

Background of Related Art

Distributed data processing is a form of information processing in which workload is performed by separate computers linked through a communication network. One form of distributed data processing which is currently widely used is dynamic workload balancing in a message driven transaction environment. Such message driven environments are described in US Patent 5,799,173, filed on May 21, 1997, and in the text, Production Workflow, Concepts and Techniques, Frank Leymann et al., published 2000, Prentice Hall, New Jersey, particularly at pp. 319-338. In a message driven transaction environment, communication between the user who initiates the transaction on a client computer and the computer systems to which portions of the transaction are dynamically distributed for performance or execution is based upon messages put into sequences of queues. A message may be considered to be a service request or a reply distributed and allocated via a server computer.

In a typical dynamic workload balancing system or process, a user on a client computer initiates a data transaction or sequence of transactions. The initiation data is sent to a server computer or a set of server

computers that will create the transactions, and which are linked to other computer systems which in turn may be individual computers or servers respectively connected to other sets of computers. A data transaction for purposes of the present description is defined as a discrete activity performed by a computer system. A transaction may be as simple as the entry of a customer order or the update of an inventory item. It may be as complex as a substantial portion of a computer program or routine. In message driven transaction environments, as in the present invention, the transactions are distributed or divided into messages which are allocated through queues and sequences of queues to linked computers or computer systems for message execution or performance. The International Business Machines Corporation (IBM) MQSeries™ application programs, e.g. the MQSeries for WindowsNT™, is described in the above Leymann et al. text at pp. 320-338. The primary advantage of workload balancing systems such as the MQSeries is that the system can operate to distribute and balance workload on computer systems operating on multiple platforms, e.g. in the case of the MQSeries, up to 35 different platforms. The workload is allocated via message queues so that the whole balancing and allocation is transparent to the initiating user and his input workload appears to be operating as if it were being done on an individual computer. All of the networking protocols in the distribution and message allocation needed to complete user input transaction requests are invisible to the user.

The advantage to the user is that the user need not be concerned with all of the networking protocols, nodes and channels which must be traversed in the workload

distribution, since this is seamlessly, automatically and dynamically done by the workload balancing algorithm. This leaves the operator/user free to concentrate on the business and production problems involved in the work. While this advantage is certainly considerable, we have found that transparentness of the workload balancing operation do provide problems to workload balancing system administrators in the case where messages are delayed or even lost in distribution and allocation. In current workload balancing systems, there is no user friendly process for tracking lost or delayed messages. In current workload balancing systems, it is necessary for system administrators to go to each queue manager's queue on any computer system that could have possibly received a message from the workload balancing algorithm, and view its list of processed or waiting messages. Even then, if the administrator is trying to track an individual message's route, the task is made particularly difficult if the message has for some reason been removed from the message queue at any particular point.

In this connection, if we conceive a workload distribution network with a hierarchy of several levels of computer systems with appropriate servers having queue manager's queues, and, in some cases, even intermediate Internet channels or connections, the difficulty of tracking becomes even more apparent.

Summary of the Present Invention

The present invention basically relates to a system, method and program for monitoring workload balancing in distributing data processing transactions into messages and dynamically allocating each of the messages to different computer systems for performance. This

comprises enabling a user to request the performance of a data processing transaction and then, without any user input, dynamically transforming via a server computer any requested transactions into messages. These messages are then allocated to different computer systems. The system has user interactive display means for displaying the allocated messages and computer systems when required by the user so that he may track the messages.

More particularly, the invention involves systems in which some of the data processing transactions are each distributed or transformed into a plurality of data processing transactions which are then allocated as described above. The invention operates effectively in workload balancing systems in which there is a server queue for storing the plurality of messages from the distributed transaction, and in which each of the different computer systems has an associated queue for storing messages allocated to each respective computer system. If one of the different computer systems has means for reallocating to other computer systems, messages initially allocated to said one computer system, then the invention provides for user interactive means for displaying the reallocated messages and computer systems to which the messages are reallocated.

Brief Description of the Drawings

The present invention will be better understood and its numerous objects and advantages will become more apparent to those skilled in the art by reference to the following drawings, in conjunction with the accompanying specification, in which:

Fig. 1 is a generalized network setup on which the workload balancing through message allocation including

the message tracking process of the present invention may be carried out;

Fig. 2 is a block diagram of a data processing system including a central processing unit and network connections via a communications adapter which is capable of functioning as the display computers and servers of Fig. 1 in carrying out the present invention;

Fig. 3 is a diagram of an interactive display screen listing the message allocations of a queue manager's queue on a primary server presented to the user tracking messages in carrying out the present invention;

Fig. 4 is the display screen of Fig. 3 as the user is selecting a message for allocation tracking;

Fig. 5 is a display screen showing the tracking history of the route of an item which the user selected from the display screen of Fig. 4;

Fig. 6 is a flowchart of an example of a typical programming process which has to be set up in a workload balancing system to implement the tracking of the present invention;

Fig. 7 is a flowchart of an illustrative run of a process set up according to Fig. 6 for message tracking; and

Fig. 8 is a flowchart setting forth an example of how the route of a tracked allocated message may be recorded so that it's status is available to previous servers in its allocation route.

Detailed Description of the Preferred Embodiment

Fig. 1 is a generalized network setup on which the workload balancing through message allocation including the message tracking process of the present invention may be carried out. By way of background, the present

invention involves the tracking of messages in message driven transaction environments wherein the user at a client terminal inputs a data processing transaction which he needs to have performed, and the transaction is dynamically transformed or distributed into messages which in turn are communicated for performance to appropriate programs on appropriate computers in a network. All this is done transparently and, thus, is invisible to the user who requested the transaction performance. In present message based transaction performance systems such as the aforementioned IBM MQSeries, message allocation for performance or execution may be seamlessly communicated across computer systems with dozens of different supported platforms. A primary premise of message-based transactions in workload allocation is that the programs involved exchange requests and responses via queues. A program sends a request to a queue rather than a specific "partnered" program. Thus, a program can put a message into a queue even if there is no program currently available to retrieve the message from the queue and process it. Queue managers manage the queues and the messages contained or manipulated in the queues. This guarantees the ultimate delivery to and execution of the messages in appropriate target queues. The primary services that message driven transaction environments require are provided by the message queue manager. Such services include administrative services to manage the queues and to manipulate the messages in the queues. In turn, the application programs which will execute the messages and reside in local or remote computer systems in the network manipulate the messages through a messaging client that provides an interface to all necessary operations even in

cross-platform computer systems. This message driven transaction environment is described in detail in the above referenced Production Workflow, Concepts and Techniques, Frank Leymann et al., particularly at pages 319-330. Background on workload balancing in a message driven transaction environment may also be found in this text and in the above-mentioned US Patent 5,799,173. It should be understood in proceeding with the following description of the present invention that the particular algorithm used in workload balancing in distributing, parsing and allocating the data transactions into messages is in no way critical to the invention. Any workload balancing algorithm may be used. The present invention is directed to the tracking of the allocated messages.

In the illustrative network of Fig. 1, a client user on computer display terminal 50 has input a transaction for performance. Terminal 50 is connected to server 51 which includes the workload balancing program and administrator which will distribute the transactions into messages and allocate the messages to particular programs in computers in the system for execution. Server 51 also includes the message queue and the message queue manager. The messages may be dynamically distributed to display computer terminals 52, 53 or 54 for performance or execution, or to other computer systems controlled by servers 65 or 61 at a next hierarchical level, e.g. the system including server 61 and display computers 62, 63 and 64. Servers 65 and 61 will respectively include their own message queues and the message queue managers. The messages may be dynamically distributed to display computer terminals 62, 63 or 64 for performance or execution, or to other computer systems controlled by

servers 75 or 71 at an even further hierarchical level, e.g. the system including server 71 and display computers 72, 73 and 74. Here again, servers 75 and 71 will in turn respectively include their own message queue and the message queue manager. In sufficiently complex distribution systems, the messages may even be allocated and communicated for execution through the World Wide Web (Web) or Internet. Since this whole allocation of messages has been dynamic and transparent to the user, the need for the message tracking system of this invention may be understood.

Referring to Fig. 2, a typical computer controlled display system is shown which may function as the computer display terminals 52-54, 62-64 and 72-74 (Fig. 1). The display system of Fig. 3 may also be used for the workload balancing and queue management servers 51, 61, 65, 71 and 75. A central processing unit (CPU) 10, may be one of the commercial PC microprocessors; when the system shown is used as the server computer, then a workstation may be used, e.g. RISC System/6000™ (RS/6000)™ series available from IBM. The CPU is interconnected to various other components by system bus 12. An operating system 41 runs on CPU 10, provides control and is used to coordinate the function of the various components of Fig. 3. Operating system 41 may be one of the commercially available operating systems such as the AIX 6000™ operating system available from IBM; Microsoft's Windows 98™ or Windows NT™, as well as UNIX™ and AIX™ operating systems. Application programs 40, in the computers performing the allocated message functions, are moved into and out of the main memory Random Access Memory (RAM) 14. In the servers, these programs would include the appropriate workload balancing algorithms,

the queue management programs, as well as the programs of the present invention for tracking the dynamically allocated messages. A Read Only Memory (ROM) 16 is connected to CPU 10 via bus 12 and includes the Basic Input/Output System (BIOS) that controls the basic computer functions. RAM 14, I/O adapter 18 and communications adapter 34 are also interconnected to system bus 12. I/O adapter 18 may be a Small Computer System Interface (SCSI) adapter that communicates with the disk storage device 20. Communications adapter 34 interconnects bus 12 into the network environment of Fig. 1 to communicate with other such computers over a Local Area Network (LAN) or through the Web or Internet. The latter two terms are meant to be generally interchangeable and are so used in the present description. I/O devices are also connected to system bus 12 via user interface adapter 22 and display adapter 36. Keyboard 24 and mouse 26 are all interconnected to bus 12 through user interface adapter 22. Display adapter 36 includes a frame buffer 39, which is a storage device that holds a representation of each pixel on the display screen 38. Images may be stored in frame buffer 39 for display on monitor 38 through various components, such as a digital to analog converter (not shown) and the like. By using the aforementioned I/O devices, a user is capable of inputting information to the system through the keyboard 24 or mouse 26 and receiving output information from the system via display 38.

Now with reference to Figs. 3 through 5, we will describe a simple illustration of how a user who has requested a data transaction on an initial input display terminal (e.g. terminal 50, Fig. 1) may track the allocation and status of the messages for carrying out

the transaction or transactions. The user on computer display terminal 50 may bring up and display the message queue listing 80, Fig. 3, of the message queue in server 51. The displayed listing shows the message ID 82, the sender 81, the status 83 of each listed message and the present stage of dynamic allocation of the message 89. As shown in Fig. 4, if the user is uncertain as to the allocation status 89 of a listed message which is the situation with the sixth queue item, he may select and, thus, highlight the item 86 and be provided with an interactive button 85 which he may then push to bring up the display of Fig. 5. This shows the allocation history 86 of the sixth queue item as it was dynamically allocated and reallocated via listed route 87 through a network such as that shown in Fig. 1. The route 87 indicates that the sixth message was received at "Angela's Machine" from which it was allocated to "Anthony's Box" where it has not been received as yet. Thus, the message is currently being worked on as Anthony's Box.

It should be noted that the accessibility of the managed messages queue as in Fig. 3 enables the interactive user to delete or modify messages listed in the queue.

With reference to Fig. 6, there will now be described an illustration of how the processes of the present invention may be set up. In any workload balancing system, such the types described above, means are set up in association with each managed queue in its respective server for recording the destination of each message sent to another queue, step 91. A process is set up so that the queue manager of each managed queue records the destination of each message sent to another

queue, step 92. A process is set up, step 93, for tracking any selected message from its original queue to its destination queue by using the queue message destination records from step 92. A process is set up, step 94, whereby each message destination queue provides the next destination for determining the route of a message until the most current destination queue is reached. A process is provided, step 95, for recording and displaying the message route determined by the process of step 94.

Now, with reference to Fig. 7, we will describe how a transaction may be processed and potentially tracked using the processes set up in Fig. 6. The user requests an initial transaction, step 101, on an input terminal, e.g. display terminal 50, Fig. 1. The workload balancing program administrator in the server dynamically distributes and/or parses the input transaction into messages and allocates messages to be sent to particular computers and computer systems for execution, step 102. The messages are put into the queue manager's queue in the server, e.g. server 51, Fig. 1. At any point in this procedure the user may request, decision step 104, a display of the queue manager's message queue. If No, the user does not make such a request, the process returns to step 102 and the allocation continues. However, if the decision from step 104 is Yes, the user has requested the display of the queue, then the queue, as shown in Fig. 3, is displayed, step 105. At this point, the user may note something in the displayed queue which may lead him to decide that he needs to view the allocation route of the message. If No, decision step 106, the process is returned to step 102, and the allocation process is continued. If the decision from step 106 is Yes, then

step 107, the selected messages route is displayed, step 107, Fig. 5. At this point, a determination may be made as to whether the session is over, decision step 108. If Yes, the session is exited. If No, then the session is again returned to step 102 and the message allocation process continued.

Fig. 8 is a flowchart setting forth an example of how the route of a tracked allocated message may be recorded (step 95 of Fig. 6) so that its status is available to previous servers in its allocation route and that route may be displayed (step 107 of Fig. 7). For purposes of describing Fig. 8, the allocation tracking display of Fig. 5 is referred to as the tracking Graphic User Interface (GUI). For tracking purposes in this example, when a workload transaction is distributed into a plurality of messages for doing the actual work, which we will refer to as "data messages", there will also be set up a plurality of "tracking messages" which will function to track the status of the data messages for the purpose of updating the tracking GUIs respectively associated with the sequence of servers in the allocation route of the data messages. Let us assume, for example, that a sending server is sending a transaction as a sequence of messages to a receiving server, which upon receiving the next message in the transaction message sequence that it is processing, step 110, determines whether it is a data message or a tracking message. If it is a data message, then a determination is made, step 113, as to whether the data message was sent from another server. If Yes, step 115, a tracking message is sent back to the sending server stating that the data message was received. In such a case, the tracking message will be received by the sending server of the data message

which now becomes the receiving server for this tracking message, and the flow will proceed through step 110 to step 112 where the tracking GUI associated with the sending server of the data message will be updated to reflect the tracking message. Also, a determination will then be made in step 117 that the tracking message came from another server, and then, step 118, another tracking message will be set back to the sending server of the first tracking message.

In any event, getting back now to decision step 113, if is determined that a data message is not from another server, the data message is sent to the queue manager, step 114, i.e. the data message may be sent to the queue manager that contains the destination queue or to the queue manager that will forward the message to the destination queue. In the case where the decision from step 113 is Yes, then the process also goes to step 114 but only after step 115 is carried out as described above by sending a tracking message back to the sending server. When this queue manager receives this data message, step 116, the server is advised through a tracking message, step 124, that the next queue manager has received the message. Then a determination is made, step 120, as to whether the data message is at its final destination. If Yes, then step 121, a tracking message is sent back to the receiving server via branch "A" to step 110 that the queue manager has received the message. If the decision is No, the data message is not at its final destination, then step 122, a tracking message is sent back to the server via branch "A" to step 110 that the data message is forwarded to another queue manager, and the data message is forwarded to the next queue manager, step 123.

A convenient implementation of the present invention is in an application program 40 made up of programming steps or instructions resident in RAM 14, Fig. 2, of the server computers during various operations. Until required by the computer system, the program instructions may be stored in another readable medium (e.g. in disk drive 20, or in a removable memory such as an optical disk for use in a CD ROM computer input, or in a floppy disk for use in a floppy disk drive computer input). Further, the program instructions may be stored in the memory of another computer prior to use in the system of the present invention and transmitted over a LAN or a Wide Area Network (WAN), such as the Internet, when required by the user of the present invention. One skilled in the art should appreciate that the processes controlling the present invention are capable of being distributed in the form of computer readable media of a variety of forms.

Although certain preferred embodiments have been shown and described, it will be understood that many changes and modifications may be made therein without departing from the scope and intent of the appended claims.